1.1 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Программа «Основы программирования на Python» разработана в соответствии с требованиями нормативных документов:

- ФЗ РФ от 29.12.2012 г. № 273-ФЗ «Об образовании в Российской Федерации»;
- Указ Президента РФ от 7.05.2012 г. № 599 «О мерах по реализации государственной политики в области образования и науки»;
- Концепция развития дополнительного образования детей до 2030 года, утверждена распоряжением Правительства РФ от 31 марта 2022 г. N 678-р;
- Приказ Министерства просвещения Российской Федерации от 09.11.2018 № 196 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам»;
- Постановление Главного государственного санитарного врача РФ от 4.07.2014 г. № 41 «Об утверждении СанПиН 2.4.4.3172-14 «Санитарно-эпидемиологические требования к устройству, содержанию и организации режима работы образовательных организаций дополнительного образования детей»;
- Методические рекомендации по проектированию дополнительных общеразвивающих программ (включая разноуровневые программы) (разработанные Минобрнауки России совместно с ГАОУ ВО «Московский государственный педагогический университет», ФГАУ «Федеральный институт развития образования», АНО ДПО «Открытое образование», 2015г.) (Письмо Министерства образования и науки РФ от 18.11.2015 № 09-3242);
- Методические рекомендации по реализации адаптированных дополнительных общеобразовательных программ, способствующих социальнопсихологической реабилитации, профессиональному самоопределению детей с ограниченными возможностями здоровья, включая детей-инвалидов, с учетом их особых образовательных потребностей. (Письмо Министерства образования и науки РФ № ВК-641/09 от 26.03.2016).

Содержание программы способствует развитию логического и алгоритмического мышления, что позволяет создавать программные продукты на основе языка программирования Python.

Уровень общеобразовательной программы: начальный.
Направленность программы: техническая.

Актуальность программы

Актуальность программы обусловлена широким распространением информационно-коммуникационных технологий в обществе и необходимостью обеспечивать связанную с этим инфраструктуру специалистами. Всё большее значение приобретает умение человека грамотно обращаться с компьютером, причём на уровне начинающего программиста.

Изучение основных принципов программирование невозможно без регулярной практики

написания программ на каком-либо языке программирования. Для обучения в рамках программы выбран язык Python, который является достаточно эффективным и доступным инструментом достижения задач в области создания программных продуктов.

Новизна программы

Python - активно развивающийся язык программирования, предназначенный для решения большого числа различных задач. Выбор Python для реализации образовательной программы обусловлен такими его преимуществами как ясность и лаконичность кода. Синтаксис языка достаточно прост и интуитивно понятен, что дает легкий порог вхождения обучающихся и позволяет сосредоточиться на алгоритмических аспектах программирования.

Отличительные особенности программы

Отличительной особенностью программы является практическая направленность.

Программа обучения предусматривает изучение основных возможностей Python, фундаментальных понятий программирования и основных алгоритмических конструкций.

В каждом модуле программы присутствует как теоретическая, так и практическая часть, поэтому учащиеся будут видеть конкретный результат освоения предмета.

Для начала обучения по Программе не требуется входного тестирования, проверяющего знания и навыки претендентов по ЯП Python.

Адресат программы

Программа ориентирована на обучающихся возрастной категории от 12 до 16 лет.

Объем и срок освоения программы

Объём программы - 72 часа (2 академических часа в неделю).

Программа рассчитана на 9 месяцев обучения.

Форма обучения: очная.

Режим занятий, периодичность и продолжительность занятий

Длительность и количество занятий – 1 раз в неделю по 2 академических часа (1 академический час равен 45 минутам).

Состав группы обучающихся – постоянный.

Количество обучающихся в одной группе: 10-12 человек.

1.2 ЦЕЛИ И ЗАДАЧИ ПРОГРАММЫ

Цели:

Освоение синтаксиса языка Python, базовых алгоритмических конструкций и формирование навыков их применения.

Задачи программы:

Обучающие:

- познакомить с синтаксисом языка программирования Python;
- формирование навыков алгоритмического и логического мышления при разработке программ;
- знакомство с принципами и методами структурного программирования;
- приобретение навыков работы в интегрированной среде разработки (IDE);
- знакомство с основными типами данных (текст, число) и типовыми методами их обработки;
- приобретение навыков разработки простых алгоритмов и программ на основе изучения языка программирования Python;
- приобретение навыков чтения чужого кода.

Развивающие:

- развитие интерес к техническим знаниям;
- развитие у обучающихся интереса к программированию;
- развитие внимательности при работе с письменными текстами;
- развитие терпения, самоконтроля;
- совершенствовать навык поиска информации в сети интернет, анализа выбранной информации на соответствие запросу, использование информации при решении задач;
- развитие логического мышления, самостоятельности, познавательного потенциала учащегося.

Воспитательные:

- развитие умения грамотно строить коммуникации;
- воспитание упорства в достижении результата;
- формирование интереса к изучению профессии, связанной с программированием.

1.3 СОДЕРЖАНИЕ ПРОГРАММЫ

Учебный план

№	Название разделов, тем	Количество часов		
		Всего	Теория	Практика
1	**Введение в программирование**	**16**	**7**	**9**
1.1	Установка Python. Среда разработки IDLE. Понятие консольной программы.	2	1	1
1.2	Ввод-вывод данных на языке Python. Переменные	2	1	1
1.3	Типы данных. Строки и числа.	2	1	1
1.4	Условный оператор.	2	1	1
1.5	Логический тип. Объединение условий.	2	1	1
1.6	Циклы. Использование цикла while	2	1	1
1.7	Цикл for	2	1	1
1.8	Решение задач на циклические алгоритмы	2	0	2
2	**Основы языка Python**	**32**	**14**	**18**
2.1	Строки и списки	2	1	1
2.2	Кортежи и словари	2	1	1
2.3	Понятие функции. Использование встроенных функций	2	1	1
2.4	Создание своих функций	2	1	1
2.5	Двумерные списки	2	1	1
2.6	Понятие модуля (библиотеки). Подключение и работа с функциями из модуля.	2	1	1
2.7	Модуль random. Работа со случайными числами.	2	1	1
2.8	Работа с файлами	2	1	1
2.9	Основы ООП. Классы и объекты.	2	1	1
2.10	Наследование	2	1	1
2.11	Создание 2D игр. Модуль PyGame	2	1	1
2.12	Спрайты. События.	2	1	1
2.13	Графика с модулем PyGame	4	1	3
2.14	Создание оконных приложений Модуль PyQt.	4	1	3
3	**Основы проектной деятельности**	**18**	**3**	**15**
3.1	Проектирование и разработка приложения «Заметки». Консольный вариант.	2	1	1
3.2	«Заметки». Оконное приложение.	4	0	4
3.3	Разработка игры с использованием модуля PyGame.	6	1	5
3.4	Разработка чат-бота для Телеграм.	6	1	5
4	**Аттестация**	**6**	**-**	**6**
	Итого:	**72**	**24**	**48**

Содержание программы

Раздел 1. Введение в язык программирования Python

Тема 1.1. Установка Python. Среда разработки IDLE. Понятие консольной программы.

Теория. Техника безопасности на занятии. Понятие «алгоритм», «исполнитель», «язык программирования», «программа», «интерпретатор». История языка программирования Python и его возможности. Виды окон в IDLE: окно программы и окно консоли. Сравнение этих окон и их возможностей.

Практика. Сохранение и запуск python-программ в среде разработки IDLE.

Тема 1.2. Ввод-вывод данных на языке Python. Переменные

Теория. Основы синтаксиса языка Python. Функция print(). Понятие «переменная». Правила именования переменных в языке Python. Оператор присваивания. Функция input().

Практика. Проект «Символьная графика». Создание определённого рисунка с помощью символов. Отработка функции print(). Проект «Попугай». В данном проекте запомнить введенную строку и несколько раз ее повторить.

Тема 1.3. Типы данных. Строки и числа

Теория. Арифметические операции с помощью математических операторов +, -, *, /. Порядок выполнения операций. Понятие «выражение», «типы данных». Функции int() и str().

Практика. Проект «Сумматор». При написании данной программы отрабатываются математические операторы и функции int() и str().

Тема 1.4. Условный оператор

Теория. Понятие «условный оператор», «вложенные команды», «оператор сравнения». Конструкция if и её синтаксис. Операторы сравнения: <, >, >=, <=, !=, ==. Структура программы. Конструкция «if-else». Команды «if» и «elif».

Практика. Решение задач на отработку условного оператора и операторов сравнения.

Тема 1.5. Логический тип. Объединение условий.

*Теория. «*True» и «False», операции «and», «or», «not». Порядок выполнения операций. Переменные без значения – «None».

Практика. Проект «Калькулятор»: создание приложения по определенным условиям.

Тема 1.6. Использование цикла «while»

Теория. Понятие «цикл с предусловием». Конструкция «while» и её синтаксис. Зацикливание и выход из цикла с помощью команды «break».

Практика. Проект «Бомба взорвалась!». Написание программы по определенным условиям.

Тема 1.7. Использование цикла for

Теория. Понятие «цикл», «цикл со счётчиком». Конструкция «for» и её синтаксис.

Практика. Проект «Таблица умножения»: создание приложения по определенным условиям.

Тема 1.8. Решение задач на циклические алгоритмы

Теория. Виды циклов и их конструкции.

*Практика. Решение задач на применение циклов «*for» и «while».

Раздел 2. Основы языка Python

Тема 2.1. Строки и списки

Теория. Понятие «строка». Создание строк. Переменные внутри строк. Операции со строками. Понятие «список». Создание списков. Добавление/удаление. Операции со списками.

Практика. Решение задач на отработку операций со строками и списками.

Тема 2.2. Кортежи и словари

Теория. Понятие «кортеж». Создание кортежа. Операции с кортежем. Понятие «словарь». Создание словаря.

Практика. Проект «Любимые вещи»: создание списка любимых развлечений и любимых лакомств.

Тема 2.3. Понятие функции. Использование встроенных функций.

Теория. Понятие «функция», «параметр функции», «значение функции». Функции: «abs», «bool», «dir», «eval», «exec», «float», «int», «len», «max», «min», «range», «sum».

Практика. Решение задач на использование математических функций.

Тема 2.4. Создание своих функций

Теория. Ключевые слова «def» и «lambda». Строение функции: «имя», «аргумент», «тело». Создание и вызов функции. Переменные и область видимости.

Практика. Решение задач на отработку понятия «функция», её строение и синтаксис.

Тема 2.5. Двумерные списки. Особенности копирования структур данных.

Теория. Вложенные списки. Особенности присваивания изменяемых и неизменяемых объектов в Питоне. Методы «copy» и «deepcopy».

Практика. Работа с таблицами.

Тема 2.6. Применение модулей

Теория. Понятие «модуль». Импортирование модуля в программу. Полезные модули: «math», «time» и другие.

Практика. Решение задач на применение модулей.

Тема 2.7. Модуль random. Работа со случайными числами.

Теория. Функции random, randint, choice и другие.

Практика. Создание игры «Угадайка».

Тема 2.8. Работа с файлами

Теория. Понятие «файл». Классификация файлов в зависимости от видов информации. Создание текстового файла. Открытие файла в программе «Python». Запись в файл.

Практика. Проект «Блокнот»: создание простейшего текстового редактора в виде приложения.

Тема 2.9. Основы ООП. Классы и объекты

Теория. Понятие «объект». Концепция объектов. Понятие «класс». Использование классов в программе «Python». Инициализация объектов.

Практика. Решение задач на отработку умения создавать классы и объекты.

Тема 2.10. Наследование

Теория. Понятие предка и потомка.

Практика. Создание структуры классов для описания животных.

Тема 2.11. Создание 2D игр. Модуль PyGame

Теория. Понятие оконной игры, главный игровой цикл. Модуль «***PyGame***» и его возможности.

Практика. Проект «Анимация»: отрисовывание объекта и программирование анимации.

Тема 2.12. Спрайты. События

Теория. Понятие «событие», «подписка на событие», «обработка события».

Практика. Проект «Игра кликер».

Тема 2.13. Графика с модулем «PyGame».

Теория. Модуль «***PyGame***» и его возможности.

Практика. Проект «Анимация»: отрисовывание объекта и программирование анимации. Проект «Фантастический мир»: рисование фигур, параметры которых задаются пользователем при запуске программы.

Тема 2.14. Создание оконных приложений Модуль PyQt

Теория. Модуль «***PyQt***». Импортирование модуля. Создание интерфейса оконного приложения.

Практика. Игра «Пятнашки»: реализация в оконном приложении.

Раздел 3. Проекты и игры на «Python»

Тема 3.1. Приложение «Заметки». Консольный вариант.

Теория. Обсуждение структуры программы: необходимые классы, способы хранения тестов. Работа с json-файлами.

Практика. Разработка программы, позволяющей создавать и сохранять заметки.

Тема 3.2. Приложение «Заметки». Оконный вариант.

Теория. Проектирование программ, допускающих дальнейшее расширение и переработку.

Практика. Добавление к разработанной программе оконного интерфейса.

Тема 3.3. Игра «Лабиринт»

Теория. Обсуждение сюжета игры: класс Лабиринт, класс Игрок и их роль. Возможные способы реализации игры: хранение уровней в коде программы или в текстовых файлах, динамическое создание уровней

Практика. Приложение-игра «Лабиринт»: Задача пользователя: дойти до конца лабиринта, 2D-игра, вид сверху. Реализация чтения и отображения уровней. Реакция на кнопки. Добавление соперников. Добавление и отображения уровня здоровья/оставшихся жизней. Экран победы и поражения.

Тема 3.4. Разработка чат-бота для мессенджера «Телеграм».

Теория. Модули в программе «Python» для создания ботов. Структура бота. Добавление сценариев диалогов и кнопок.

Практика. Написание проекта «Тесты» в виде бота.

Раздел 4. Аттестация

Практика. Разработка собственного приложения. Создание презентации и подготовка выступления. Защита проекта.

1.4 ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ

Предметные результаты.

- Умение разбивать решение задачи на подзадачи, составлять алгоритм;
- Иметь представление о таких типах данных, как число, текст, и уметь применять типовые методы при работе ними.
- Знать правила построения условных конструкций, в том числе сложных и вложенных;
- Знать правила построения циклических конструкциях, отличия циклов со счётчиком от циклов с предусловием, способы прерывания и продолжения циклов;
- Умение объяснять и использовать на практике как простые, так и сложные структуры данных и конструкции для работы с ними;
- Умение искать и обрабатывать ошибки в коде;
- Умение анализировать как свой, так и чужой код;
- Умение импортировать модули в программу;
- Умение работать с вычислительной техникой;
- Знание основных классических алгоритмов и способы их реализации;
- Знание основных элементы программирования;
- Умение создавать оконные приложения и 2D игры.

Метапредметные результаты

- Умение создавать модели для решения практических задач;
- Умение разрешать конфликты на основе выработки общей позиции;
- Умение критически оценивать сроки реализации задуманного проекта;
- Умение вносить изменение в проект, корректировать изначальный план;
- Умение ставить для себя новые задачи;
- Умение работать с информацией: находить, оценивать и использовать информацию из различных источников, необходимую для решения профессиональных задач (в том числе на основе системного подхода);
- Развитие навыков смыслового чтения и структурирование информации.

Личностные результаты.

- Формирование навыка доводить дело до конца
- Формирование мировоззрения, необходимого в современном цифровом мире.
- Формирования позитивного отношения к обучающимся, их мнению.
- Формирование ответственного отношения к обучению.
- Появление чувства ответственности за проделанный труд, удовлетворенности от результатов труда;
- Умение грамотно строить коммуникацию, исходя из целей и ситуации.

2. КОМПЛЕКС ОРГАНИЗАЦИОННО-ПЕДАГОГИЧЕСКИХ УСЛОВИЙ

2.1 КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК

Период обучения 9 месяца, 1 раз в неделю. Всего занятий – 36.

№	Тема занятия	Деятельность
1	Установка Python. Среда разработки IDLE. Понятие консольной программы.	КЗ
2	Ввод-вывод данных на языке Python. Переменные	КЗ
3	Типы данных. Строки и числа.	КЗ
4	Условный оператор.	КЗ
5	Логический тип. Объединение условий.	КЗ
6	Циклы. Использование цикла while	КЗ
7	Цикл for	КЗ
8	Решение задач на циклические алгоритмы	П
9	Строки и списки	КЗ
10	Кортежи и словари	КЗ
11	Понятие функции. Использование встроенных функций	КЗ
12	Создание своих функций	КЗ
13	Двумерные списки	КЗ
14	Понятие модуля (библиотеки). Подключение и работа с функциями из модуля.	КЗ
15	Модуль random. Работа со случайными числами.	КЗ
16	Работа с файлами	КЗ
17	Основы ООП. Классы и объекты.	КЗ
18	Наследование	КЗ
19	Создание 2D игр. Модуль PyGame	КЗ
20	Спрайты. События.	КЗ
21	Графика с модулем PyGame	КЗ
22	Графика с модулем PyGame – продолжение.	П
23	Создание оконных приложений Модуль PyQt.	КЗ
24	Создание оконных приложений Модуль PyQt – 2 часть.	ПР
25	Проектирование и разработка приложения «Заметки». Консольный вариант.	КЗ
26	Оконное приложение «Заметки».	ПР
27	Оконное приложение «Заметки» - 2 часть.	ПР
28	Разработка игры с использованием модуля PyGame.	КЗ
29	Разработка игры – 2 часть	ПР
30	Разработка игры – 3 часть	ПР
31	Разработка чат-бота для Телеграм.	КЗ
32	Разработка чат-бота для Телеграм – 2 часть	ПР
33	Разработка чат-бота для Телеграм – 3 часть	ПР
34	Доработка собственного приложения, групповые обсуждения.	ПР
35	Создание презентации и подготовка выступления.	ПР
36	Защита проекта.	ПР

Календарный учебный график заполнен с помощью условных обозначений:

- КЗ – комбинированные занятия, сочетающие элементы теории и практики;
- П – практикум;
- ПР – проектная работа (работа над кейсами).

2.2 УСЛОВИЯ РЕАЛИЗАЦИИ ПРОГРАММЫ

Материально-техническое обеспечение программы:

- персональные компьютеры (на каждого обучающегося) с программным обеспечением, с минимальными системными требованиями (процессор i3 или аналог, 4 Гб оперативной памяти)
- экран, проектор;
- установленный на каждый компьютер интерпретатор языка программирования Python 3.8 или новее, с установленным IDE PyCharm.

Методы и приемы работы

Методы обучения:

- объяснительно-иллюстративный (беседы, объяснения, дискуссии);
- репродуктивный (деятельность обучаемых носит алгоритмический характер, выполняется по инструкциям, предписаниям, правилам в аналогичных, сходных с показанным образцом ситуациях);
- метод проблемного изложения;
- эвристический (метод обучения заключается в организации активного поиска решения выдвинутых в обучении (или самостоятельно сформулированных) познавательных задач в ходе подготовки и реализации творческих проектов);
- исследовательский.

Формы организации учебного занятия:

- *вводное занятие* – педагог знакомит обучающихся с техникой безопасности, особенностями организации деятельности и предлагаемым планом работы на текущий год;
- *ознакомительное занятие* – педагог знакомит обучающихся с новыми методами работы в зависимости от темы занятия;
- *тематическое занятие* – на котором детям предлагается работать над моделированием по определенной теме. Занятие содействует развитию творческого воображения обучающихся;
- *занятие-проект* – на таком занятии обучающиеся получают полную свободу в выборе направления работы, не ограниченного определенной тематикой. Обучающиеся, участвующие в работе по выполнению предложенного задания, рассказывают о выполненной работе, о ходе выполнения задания, о назначении выполненного проекта;
- *конкурсное игровое занятие* – строится в виде соревнования для повышения активности обучающихся и их коммуникации между собой;
- *комбинированное занятие* – проводится для решения нескольких учебных задач;
- *итоговое занятие* – служит подведению итогов работы за учебный год, может проходить в виде мини-выставок, просмотров творческих работ и презентаций.

Алгоритм комбинированного учебного занятия:

1. Мотивация обучающихся.
2. Актуализация имеющихся знаний.
3. Теоретический блок нового материала.
4. Закрепление материала.
5. Перерыв.
6. Теоретический блок нового материала.
7. Закрепление материала.
8. Рефлексия.

2.3 ФОРМЫ АТТЕСТАЦИИ

Формы оценки уровня достижений обучающегося

Для контроля и самоконтроля за эффективностью обучения применяются методы:

- предварительные (наблюдение, опрос);
- текущие (наблюдение);
- тематические (контрольные вопросы, промежуточные задания);
- итоговые (мини-проект).

Формы фиксации образовательных результатов

Для фиксации образовательных результатов в рамках курса используются:

- Промежуточная аттестация проводится в конце первого полугодия.
- Итоговый контроль проводится в конце года с целью определения степени достижения результатов обучения и получения сведений для совершенствования программы и методов обучения;
- отзывы обучающихся по итогам занятий и итогам обучения.

Формы предъявления и демонстрации образовательных результатов:

- защита мини-проекта.

Формы подведения итогов реализации программы

- Педагогический мониторинг позволяет систематически отслеживать результативность реализации программы. Мониторинг включает в себя традиционные формы контроля: промежуточную и итоговую аттестацию результатов обучения детей.
- Аттестация обучающихся может проходить на итоговом занятии в форме презентации своего проекта.

2.4 ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

Оценивание развития учащихся проводится на основе следующего перечня компетенций:
Технические:

- анализ данных
- алгоритмическое мышление
- программирование и написание кода

Гибкие:

- критическое мышление
- работа в коллективе, эффективная коммуникация
- презентация проекта с точки зрения социального воздействия.

Текущий контроль сформированности результатов освоения программы осуществляется с помощью следующих инструментов:

- на каждом занятии: опрос, написание кода в интерпретаторе языка программирования Python, самоконтроль ученика;
- на уроках-практикумах: написание кода в интерпретаторе языка программирования Python, взаимоконтроль учеников, самоконтроль ученика;
- при выполнении проектов: написание кода в интерпретаторе языка программирования Python, презентация проекта.

Критерии оценивания выполнения практических заданий

Оцениваемый результат	**Недостаточный уровень**	**Средний уровень**	**Высокий уровень**

Владение навыков алгоритмического мышления и понимание необходимости формального описания алгоритмов	Обучающиеся не способны определить подходящую алгоритмическую конструкцию для формального описания алгоритма решения практической задачи	Обучающиеся способны определить подходящую алгоритмическую конструкцию для формального описания алгоритма решения практической задачи при помощи преподавателя	Обучающиеся способны самостоятельно определить подходящую алгоритмическую конструкцию для формального описания алгоритма решения практической задачи
Навыки применения таких структур данных, как число, текст	Обучающиеся не способны определить разницу между указанными типами данных, смысл их использования при решении конкретных задач	Обучающиеся способны определять при помощи преподавателя типы данных, необходимых для реализации алгоритма решения задачи	Обучающиеся способны самостоятельно определять типы данных, необходимых для реализации алгоритма решения задачи
Навыки построения условных конструкций	Обучающиеся не способны описать, как работает ветвление, его предназначение. Обучающийся не может самостоятельно использовать условные конструкции для решения практической задачи.	Обучающиеся способны описать, как работает ветвление, его предназначение. Обучающийся может с помощью преподавателя использовать условные конструкции для решения практической задачи.	Обучающийся может самостоятельно использовать условные конструкции для решения практической задачи, определить работоспособную форму построения ветвления.
Навыки построения циклических конструкций	Обучающиеся не способны описать, как работает цикл, его предназначение. Обучающийся не может самостоятельно использовать циклические конструкции для решения практической задачи.	Обучающиеся способны описать, как работает цикл, его предназначение. Обучающийся может с помощью преподавателя использовать циклические конструкции для решения практической задачи.	Обучающийся может самостоятельно использовать циклические конструкции для решения практической задачи, определить работоспособную форму построения цикла.

Навыки разработки программ и их отладки	Обучающийся не может самостоятельно применить пошаговую отладку, найти места ошибок в программном коде, внести корректные изменения и разработать корректную программу.	Обучающийся может самостоятельно применить пошаговую отладку, найти места большинства ошибок в программном коде. С помощью преподавателя может внести корректные изменения в программный код и разработать корректную программу.	Обучающийся может самостоятельно написать программный код, применить пошаговую отладку, найти места ошибок, внести корректные изменения в программный код.
Навыки чтения чужого кода	Обучающийся не может самостоятельно изучить код программы, понять логику разработки алгоритма, методы, применённые в программе, выявить места ошибок, внести изменения для корректировки программы.	Обучающийся может самостоятельно изучить код программы и выявить методы, применённые в программе. С помощью преподавателя понять логику разработки алгоритма, выявить места ошибок, внести изменения для корректировки программы.	Обучающийся может самостоятельно изучить код программы, понять логику разработки алгоритма, методы, применённые в программе, выявить места ошибок, внести изменения для корректировки программы, предложить способы её совершенствования.
Выполнение итогового мини-проекта «Текстовый лабиринт»	Обучающийся не может запрограммировать текстовый лабиринт без примера кода.	Обучающийся может запрограммировать текстовый лабиринт с минимальными подсказками и примерами кода.	Обучающийся может запрограммировать текстовый лабиринт со множеством дополнительных разветвлений и функций (например, телепорт, сбор предметов и т.д.) с минимальными подсказками и примерами кода.
Навыки коммуникации и презентации	Недостаточная уверенность, аргументация позиций	Уверенность во время выступления, хороший стиль речи, аргументированность и убедительность. Хорошая	Уверенность во время выступления, отличный стиль речи, высокая убедительность и аргументированность.

		визуализация защиты	Качественная визуализация защиты

Для оценивания проекта, по созданию лабиринта, заполняется таблица с критериями, за каждый из которых дается определенное количество баллов. Основные критерии, по которым выставляются баллы:

1) соответствие проекта заданию (0-2 балла);

2) творческий подход (0-3 баллов);

3) сложность проекта (0-5 баллов);

4) качество алгоритмов (0-10 баллов);

5) отсутствие ошибок в проекте (0-5 баллов);

6) качество презентации — содержательность, логичность, креативность представления проекта (0-5 баллов).

Баллы суммируются, и на основании этого делается заключение об уровне сложности и успешности выполненного проекта

Общая сумма:

14 баллов и меньше – низкий уровень освоения программы;

15-23 баллов – базовый уровень освоения программы;

24 – 30 баллов– высокий уровень освоения программы.

Результаты итогового контроля заносятся в таблицу (приложение 1).

2.5 МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ

В качестве методов обучения по программе используются словесный, наглядный, практический, проблемный, проектные методы.

В качестве методов воспитания по программе используются упражнение, убеждение, мотивация, поощрение.

На занятиях используются различные формы организации образовательного процесса:

- индивидуальная;
- групповая.

Формы организации учебного занятия:

- лекции;
- практические занятия;
- защита мини-проекта.

3. СПИСОК ЛИТЕРАТУРЫ

Список литературы для педагога:

1. Закон РФ «Об образовании».
2. Поляков К.Ю. Программирование. Python. C++. В 4 ч.: учебное пособие/К.Ю.Пляков. – М.:Бином. Лаборатория знаний, 2019.
3. Прохоренок Н. А., Дронов В.А. Python 3. Самое необходимое – СПб: БХВ-Петербург, 2019.
4. Мэтиз Э. Изучаем Python. Программирование игр, визуализация данных, веб-приложения //Пер. с англ.: учебник/Э. Мэтиз. 2-е изд. – Спб.: Питер. – 2018.
5. Лутц М. Python. Карманный справочник. Пер. с англ. //М.: ИД Вильямс. – 2019.
6. Бизли Д.М., Г. Ван Россум. Язык программирования Python. Справочник. (пер. с англ.) Киев: ДиаСофт., 2000.
7. Васильев Денис Алексеевич Методические особенности изучения языка Python школьниками // Символ науки. 2017. №1. URL: https://cyberleninka.ru/article/n/metodicheskie-osobennosti-izucheniya-yazyka-python-shkolnikami (дата обращения: 15.01.2019).

Список литературы для обучающихся

1. ПИТОНТЬЮТОР URL: https://pythontutor.ru (дата обращения: 19.08.2022).
2. Python: основы и применение // Stepik URL: https://stepik.org/course/512/ (дата обращения: 19.08.2022).
3. "Поколение Python": курс для начинающих// Stepik URL: https://stepik.org/course/58852/ (дата обращения: 19.08.2022)